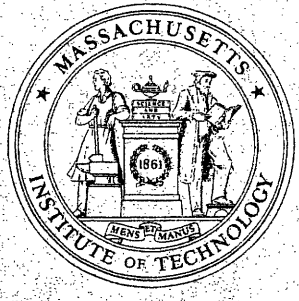
OPERATIONS RESEARCH CENTER

working paper



MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Efficient Reliability Computation of Some Highly Structured Networks

by

Daniel Bienstock

OR 142-85 October 1985

The research reported in this paper was carried out at MIT and was partially funded by grant NSF-ECS-8316224.

Abstract

We consider several classes of special networks with relevant asymptotic properties, whose reliability can be computed in polynomial time.

Efficient Reliability Computation of Some Highly Structured Networks

Daniel Bienstock, GSIA, Carnegie-Mellon University

1. Introduction

The problem of computing the probability that a graph (or a given subset of the nodes of a graph) remains connected, while its arcs are independently erased with known probabilities, is #P-hard [2], and as a consequence, a formidable task. Not surprisingly, the best algorithm (to date) has complexity strictly exponential in the total number of nodes of the graph (Buzacott [6]).

These negative results refer to arbitrary graphs. In some applications, however, one would encounter certain topologies only. The standard network reliability literature has successfully treated the case of trees, series-parallel graphs [1] (in the directed case), and other graphs with simple topologies. All these graphs share a common element: they have a simple local structure, even though their global structure may be complex, and their reliability analysis requires a representation of the entire graph.

The graphs we consider in this paper are intended for very large-scale applications, such as power or ground networks of large-scale circuit boards or VLSI chips. Our graphs are constructed by piecing together several copies of a building block graph, in a highly structured manner (in fact, the resulting graph may be exponentially larger than the building block). Thus, their topology, while complex from a local point of view, has some nontrivial (e.g., recursive) global properties. We will refer to such graphs as *composite* graphs.

In order to represent and analyze a composite graph, we only need a representation of the

building block, together with some (usually recursive) rules to obtain the overall graph. The advantages of composite graphs over arbitrary graphs are considerable from the point of view of design, construction and operation. These, of course, are ideas that have long been exploited in the VLSI theory literature. The specific graphs that we consider in this paper have two additional attractive features: their reliability can be computed in polynomial time, and as detailed in Section 6, they are asymptotically optimal from a reliability vs. cost point of view, since they can be designed to attain high reliability with minimum number of arcs. We describe our composite graphs in Section 2.

Our algorithms utilize Buzacott's [6] as a subroutine, together with some common ideas from the algorithms of Rosenthal [10] and Fratta and Montanari [9], and some lattice-theoretic tools.

2. Description of Networks and Computational Bounds

Consider an arbitrary graph L with m nodes. L contains two disjoint distinguished subsets of nodes, S and T, both with d nodes. The nodes of S are indexed 1,2,...,d, and similarly with T. We are especially interested in the cases in which L is dense and $d \ll m$.

We will build more complex structures by taking many copies of L and plugging them into each other using the sets S and T. The simplest such graph is obtained as follows: take k copies of L, numbered 1,2,...,k, and for $1 \le i \le k-1$, identify the set S in copy i with the set T in copy i+1 indexwise(see Figure 1(b)). Let G_k denote the resulting composite graph. G_k has $n = km-(k-1)d \approx km$ nodes. Notice that in order to represent G_k, we only need a representation of

L, together with an "underlying graph" which is a linear chain on k nodes: each node represents a copy of L, and the arcs tell us how these copies are interfaced.

Figure 1 - (a)Graph L. (b)Graph G_k. (c)Underlying graph.

Now suppose that

$m \leq x\log n$, and $d \leq y\log n/\log\log n$,

where x and y are constants (this corresponds to the case where $k=O(n/\log n)$ and k is exponentially larger than m). Then, as shown in Section 5, we can compute the probability that G_k remains connected in time (at most)

$$n^{x\log 3 + 2y} + n^{3y}.$$

A very special case arises when L has some symmetry properties. For example, suppose L is invariant under arbitrary permutations of the set S (the invariance is both from a graph theoretic sense and with respect to the arc failure distribution). In such cases, the parameter d can be larger: if

$d \leq y\log n$,

and m is as above, we can compute the reliability of G_k in time (roughly)

$$n^{x\log 3} + n^{y\log 3}.$$

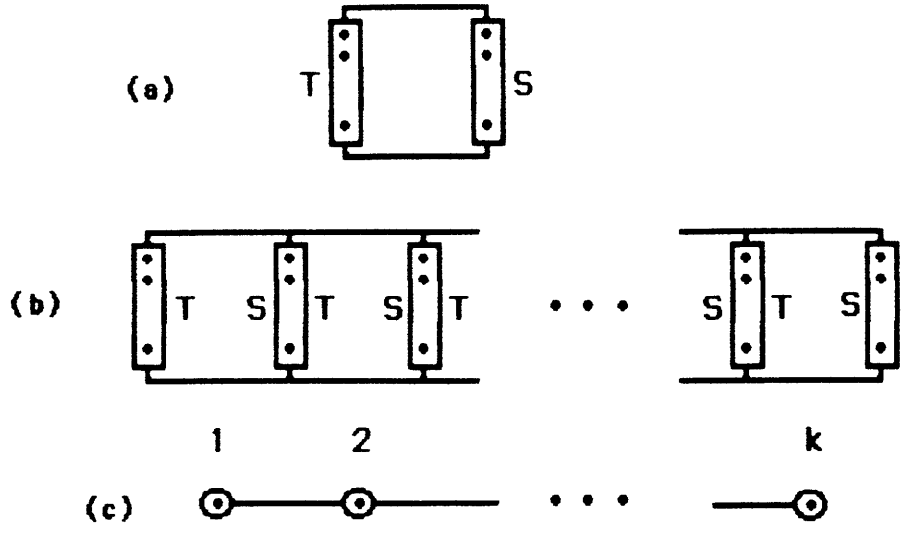


FIGURE 1

Using the underlying graph concept, we can also describe more complex networks that can be analyzed in polynomial time. An important case is that of complete k-ary trees, where k is arbitrary (it could depend on n) and also variations thereof, such as the one shown in Figure 2.

Figure 2 - Variation of complete k-ary tree, shown for k=3.

In fact, the underlying graph can be an arbitrary tree, and we will still retain polynomial complexity (in such cases the sets S and T may be partitioned into smaller subsets in order to achieve all the connections in the underlying graph). The most general classes of graphs for which we are able to claim polynomial running time can be described as follows:

There is a constant s and a recursive decomposition scheme of the underlying graph U, such that every subgraph X of U produced in the decomposition has at most s nodes adjacent to U-X.

We will call such underlying graphs s-separable (for example, trees are 1-separable, and the graph in Figure 2 is 2-separable).

Finally, we can even use <u>different</u> building blocks instead of a single graph L and still obtain polynomial complexity. In the linear chain case, the two extreme blocks could be different from the inner ones, and in the complete k-ary case, a different block could be used at each level. Nevertheless, once more, we are mainly interested in the cases where a single block is used.

Some further generalizations are outlined in Section 7.

Our algorithms, roughly, proceed as follows. First, we carry out "local" enumeration by

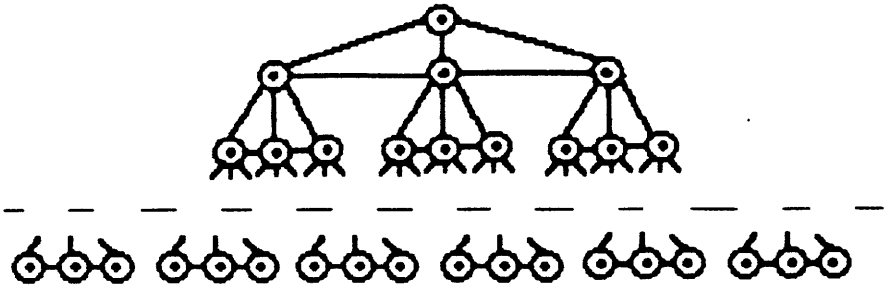


FIGURE 2

analyzing graph L in various ways, using Buzacott's algorithm as a subroutine. Then, by utilizing ideas from the Rosenthal, Fratta and Montanari algorithms and exploiting the special structure of the underlying graph, we set up a set of homogeneous systems of linear difference equations, whose solution will yield (among other parameters) the desired reliability. The transition from the local analysis to the global analysis is made via certain lattice-theoretic tools, outlined in Section 4.

3. Brief Review of the Algorithms of Buzacott, Rosenthal, Fratta and Montanari

The algorithm of Buzacott computes the probability that a given subset of nodes of a graph remains connected under independent arc failures. This algorithm may be more properly regarded as a general counting technique, rather than a graph theoretic procedure. Not surprisingly, it appears to be most efficient when applied to very dense graphs, and fairly inefficient otherwise. Yet, this algorithm has the best provable upper bound on complexity, 3^n, where n = total number of nodes.

The algorithms of Rosenthal, and Fratta and Montanari are radically different from Buzacott's. These two algorithms share one basic (very important, in our view) idea, which we next describe with our notation.

Let H be a graph whose arcs are erased independently, and E a w-node subset of the nodes of H. Let x be a partition of E with j blocks. Denote by P(x) the probability that H is split into exactly j connected components, each containing a different block of x. We will call this probability an E-bond probability of H, and the event whose probability is P(x) an E-bond event of H (the event will be represented by x). The total number of such events is at most

B(w), the number of partitions of a w-element set, a quantity that grows nearly as rapidly as w! [7].

The algorithms of Rosenthal, Fratta and Montanari compute bond probabilities recursively. In order for these algorithms to be efficient, it is crucial to find a recursive decomposition of the original graph that requires small node cuts only. Consequently, these algorithms are quite inefficient when applied to dense graphs, or graphs with dense subgraphs (i.e., almost exactly the opposite of Buzacott's) [4]. In the next section we will provide a different way of computing bond probabilities that is intended for dense graphs.

There is another common idea to [9] and [10]. Let H be a graph, and C a node cut that splits H into H_1 and H_2. Suppose we condition on the occurrence of a C-bond event of H_1 corresponding to a certain partition x. Since each block of x is connected via H_1, we can then analyze H_2 by collapsing each block of x into a single node.

4. Some Lattice-Theoretic Tools.

Let H be an m-node graph with a special w-node subset E, as in the previous section. There is a lattice-theoretic method of computing the E-bond probabilities which is more efficient than the Rosenthal, Fratta and Montanari method whenever H is dense and $w \ll m$. This method is described in [4], and in this paper we will summarize the relevant results. First we introduce some notation.

Given a partition y of E, form graph H[y] by collapsing each block of y into a single node. Let R[y] denote the probability that H[y] remains connected, and R the B(w)-length vector of

entries R[·]. Then

$$R = A(w)P, \qquad (1)$$

where A(w) is a 0-1 matrix that depends only in w, and that is invertible for all w. In fact, relationship (1) holds for arbitrary arc failure distributions. For more background on (1) and related topics the reader is referred to [4]. Consequently, in order to compute P, we may first compute R and then solve system (1). In case of independent arc failures, we can use Buzacott's algorithm to compute the vector R, and the overall procedure (including the generation of matrix B(w)) will then have complexity

$$O(B(w)3^m + B^3(w)). \qquad (2)$$

If the graph H is invariant under permutations of E, system (1) is somewhat "redundant." For example, suppose that w=6. If the invariance holds, then R[123,45,6]=R[246,35,1] and similarly P(12,3456)=P(35,1246). In fact, whenever two partitions x and y have the same number of blocks of each cardinality (in lattice-theoretic language, if x and y are of the same *class*), then R[x]=R[y] and P(x)=P(y). Thus, system (1) may be abbreviated by selecting only one partition per class, obtaining two vectors R' and P' from R and P, and similarly, replacing A(w) with a suitable matrix which we denote by A(2,w). That is, A(2,w) is obtained from A(w) by selecting one row per partition class, and replacing all columns corresponding to each partition class by their sum. In that case, we will have

$$R' = A(2,w)P'. \qquad (1')$$

As matrix A(2,w) is also invertible [4], we can solve for P' from R'. Matrix A(2,w) can be generated directly (i.e., without generating A(w) first), in which case the computation of P' will take

$$O(3^m p(w) + 3^w p^2(w) w), \qquad (2')$$

where $p(w)$ is the number of partition classes of w elements [4]. Complexity (2') is considerably smaller than (2) since $p(w)$ grows strictly exponentially in $\sqrt{w}$ (i.e., much slower than $B(w)$).

To conclude this section, we present some standard lattice-theoretic notation as it applies to bond events. Let $H(1)$ and $H(2)$ be two arc disjoint graphs which share a subset of nodes S. Let $x(i)$ be an S-bond event of $H(i)$, for $i=1,2$. If both $x(1)$ and $x(2)$ simultaneously occur, the nodes of S will be split into connected components corresponding to the blocks of a certain partition z. In fact, in lattice-theoretic language, $z = x(1) \wedge x(2)$, where the $\wedge$ operator is called the dual meet. Given $x(i)$, $i=1,2$, their meet can easily be computed in time linear in $|S|$.

5. Polynomial-Time Reliability Algorithms for Composite Networks

We will first describe how to compute the reliability of composite graph G_k depicted in Figure 1, in the general (non-symmetric) case.

This problem will be embedded in a family of problems. Using the notation of the previous section, given a partition y of the set T, let $L[y]$ be the graph obtained from L by coalescingg the blocks of T according to y, and $G_k[y]$ the graph obtained from G_k by replacing the leftmost copy of L in G_k with $L[y]$. Finally, let $R_k[y]$ denote the reliability of $G_k[y]$ (notice that if y is the partition with exactly d blocks of one element each, then $G_k[y]=G_k$), and R_k the vector of entries

$R_k[\cdot]$. There is a simple recursion satisfied by R_k, as follows.

For partitions x and y respectively of T and S, let $p(x,y) = R[x](y)$ = S-bond probability of y in L[x]. Also, let M denote the B(d)xB(d) matrix whose (x,y)-entry is p(x,y). Then

$$R_k = M R_{k-1}. \qquad (3)$$

Each row of (3) follows by conditioning on all S-bond events y of S. Consequently,

$$R_k = M^{k-1} R_1, \qquad (4)$$

where the x-entry of vector R_1 is the connectivity probability of graph L[x].

In order to compute each row of matrix M, we solve a system of the form (2), R=A(d)P (but we only need to generate and invert matrix A(d) once). Thus, matrix M can be computed in time

$$O(B^3(d) + 3^m B^2(d)).$$

Also, M can be raised to the power k-1 in O(logk)=O(logn) matrix multiplications (this standard trick follows by using the binary expansion of k-1). Finally, notice that for each x, $R_1[x]$ will be computed in order to find the probabilities p(x,·) (i.e., in order to solve system (2) to find the S-bond probabilities of L[x]). Therefore, we can compute vector R_k in time

$$O(3^m B^2(d) + \log n\, B^3(d)),$$

which is less than

$$n^{x\log 3} + 2y + n^{3y} \qquad (5)$$

if $m \le x\log n$, and $d \le y\log n/\log\log n$ (see [4] or [7] for a detailed analysis of the function B).

A similar result will hold in the symmetric case, i.e. when L is invariant under

permutations of S and T. The system of difference equations to be solved is similar to (4), but the size of the system is p(d), rather than B(d) (and as a result, we can allow d to be proportional to logn while preserving polynomial complexity). The reader is referred to [4] for more details.

Finally, we point out that if instead of using a single building block L, composite graph G_k is constructed with k different building blocks, we will obtain a different recursion (3) for each k. The total workload will be, at most,

$$n^{1+x\log 3+2y} + n^{3y}.$$

We will next sketch how to compute the reliability of a composite network whose underlying graph is a complete k-ary tree. When constructing such a network, the sets S and T are used, respectively, to connect each copy of L to its father and sons (in the underlying tree). The copy of L corresponding to the root in the tree will also be called the root of the composite graph.

More formally, set $G_{k,1}=L$, and denote by S_1 the copy of S in $G_{k,1}$. Next, for $h>1$, let S_{h-1} denote the copy of S in the root of $G_{k,h-1}$. We take k copies of graph $G_{k,h-1}$, and indexwise simultaneously identify their k copies of set S_{h-1} into a set S'_h, to obtain graph $G'_{k,h}$. Finally, to obtain $G_{k,h}$ we take an additional copy of L and indexwise identify its copy of T with set S'_h. The new copy of L will be the root of $G_{k,h}$, and its copy of S will be denoted by S_h.

Figure 3 - Construction of $G_{k,h}$

We embed the problem of computing the reliability of $G_{k,h}$ in the larger problem of computing the vector P_h of S_h-bond probabilities of $G_{k,h}$. To solve this problem we iteratively compute P_i, for increasing values of i=1,2,...,h. Notice that P_1 is the set of S-bond probabilities of L.

Assuming we have already computed P_{h-1}, to find P_h via relationship (2) (or (2') in case of symmetry) we first compute all reliabilities $R_h[z]$ of graph $G_{k,h}[z]$, where we coalesce the nodes of S_h according to partition z (see Figure 4(a)). Each of these parameters is computed by solving a system of difference equations, as follows.

We can compute the S'_h-bond probabilities of $G'_{k,h}$ by solving a system reminiscent of (4): let $P'_{j,h}$ denote the vector of S'_h-bond probabilities of $G'_{j,h}$, for $1 \le j \le k$. Notice that $G'_{1,h} = G_{k,h-1}$ and therefore $P'_{1,h} = P_{h-1}$. Let N_h be the B(d)xB(d) matrix whose (x,y)-entry is

$$\sum P_h(w),$$

the sum being taken over all partitions w of S such that $w \wedge y = x$. Then

$$P'_{k,h} = N_h P'_{k-1,h} = N_h^{k-1} P'_{1,h} = N_h^{k-1} P_{h-1}. \qquad (6)$$

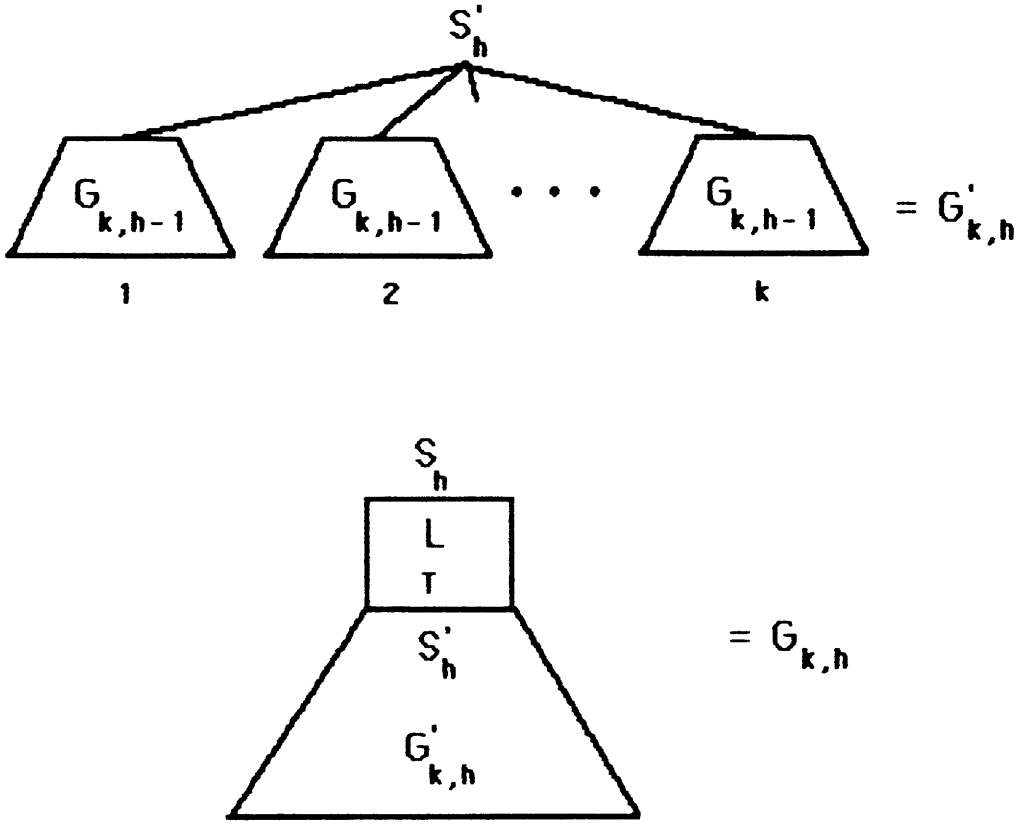


FIGURE 3

Having computed $P'_{k,h}$, we then compute $R_h[z]$, for any given z, by

$$R_h[z] = \sum P'_{k,h}(x)\, R[x,z] \qquad (7)$$

where the sum is taken over all partitions x of S, and R[x,z] is the connectivity probability of the graph obtained from L by coalescing nodes of S according to z and nodes of T according to x (see Figure 4(b)).

Figure 4 - (a) Graph $G_{k,h}[z]$. (b) Depiction of equation (7).

Finally, we have

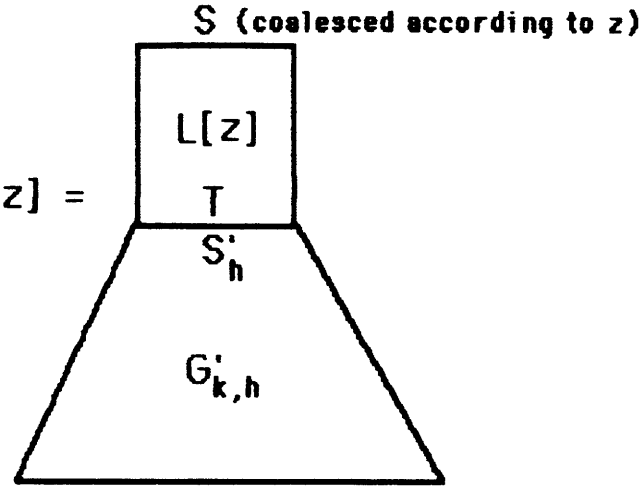
$$P_h = A(d)^{-1} R_h. \qquad (8)$$

Equation (8) is the final step in the analysis of $G_{k,h}$.

It is possible to show that the total workload in the analysis of $G_{k,h}$ is larger by a factor of B(d) from that of the linear chain case (and therefore it is polynomial in n under our assumptions concerning m and d). The reader is referred to [1] for additional details.

Next, we state how to extend the analysis for the cases in which the underlying graph is an arbitrary tree. Let us return for a moment to the complete k-ary tree case, and suppose that

(a) $G_{k,h}[z] =$



(b)

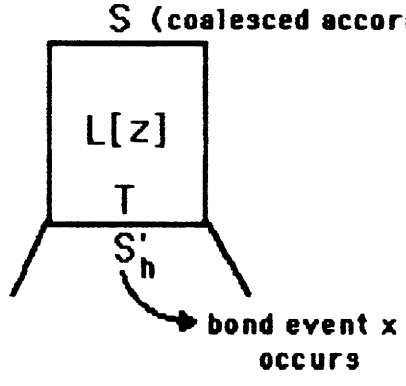


FIGURE 4

in constructing graph $G_{k,h}$ we use k different graphs as the sons of the root, instead of k identical copies of $G_{k,h-1}$. Assuming we have already analyzed the graph corresponding to each son, we can analyze $G_{k,h}$ essentially as in the preceding paragraphs. The main difference will be that instead of (6), we will have an equation of the form

$$P'_{k,h} = N_{h,k} N_{h,k-1} \cdots N_{h,2} P_{h-1}, \qquad (6')$$

where the $N_{h,q}$'s are matrices of the same form as N_h in (6). This type of reasoning can be applied whenever we use <u>different</u> building blocks to construct our composite graph, rather than a single building block L. In this case, we separately analyze each rooted subtree (consisting of a node and all its descendants) rather than a single subtree of each height, via an equation of the form (6'). This will result in an increase of complexity by a factor of n/m, from the identical building block case.

Moreover, the number of sons of a node in the underlying tree can be different for each node (i.e., not necessarily a constant k), and essentially the same procedure as above will efficiently compute reliability, by analyzing all rooted subtrees from the "bottom up" and using equations (6') and (7). Finally, notice that any tree can be drawn as a rooted tree.

* * *

The remaining cases of underlying graphs are similarly analyzed. The common feature to all these algorithms is that we extensively analyze the building block (or blocks) and then utilize the structural properties of the <u>underlying graph</u> to extend the analysis to the overall

graph. It is essential that we utilize Buzacott's algorithm (or a better one, if available) to analyze the building block, rather than the Rosenthal, Fratta and Montanari approach: if the number of nodes in the building block grows as logn, the latter approach will be nonpolynomial in n [4], in the worst case.

6. Asymptotic Relevance of Composite Networks

Suppose that the building block graph L is dense, and that m grows as logn (as in the previous sections). It follows that the composite network has O(nlogn) arcs. Further, assume that all the arcs are equally reliable (or nearly equally reliable). Then we can build our composite networks to achieve arbitrarily high reliability (converging to 1 as n grows to infinity) [3].

On the other hand, still under the constant arc reliability assumption, any growth in the number of arcs slower than nlogn will force system reliability to be arbitrarily low (converging to 0 as n grows to infinity) [3], [8].

Thus our composite networks are asymptotically "optimal" or "efficient." Of course, there are other classes of optimal networks. However, many of those networks either have a complex or nonstructured underlying topology, which makes them unattractive (for instance, many global connections would be unattractive).

Finally, from a practical standpoint, in building a composite network of the kind considered in this paper, a designer would ideally wish to maintain the parameter m as low as possible (i.e., in order to preserve as simple a local structure as possible). The above results state that m must grow at least as fast as logn. Hence, it is meaningful to seek polynomial-time

reliability algorithms for composite graphs with m=O(logn).

7. Some Extensions

We can improve on our algorithms whenever graph L has some special properties. For example, if L is complete, or complete J-partite (for some constant J), an implicit implementation of Buzacott's algorithm will have complexity polynomial in m. In such cases we can efficiently analyze composite networks with m growing as fast as n (however, we must retain $d=O(\log^2 n)$ so that p(d) is polynomial in n).

Similarly, if L is planar, it can be analyzed in time strictly exponential in $\sqrt{m}$ [5]. Thus we can have $m=O(\log^2 n)$ and preserve polynomial complexity (we still retain d=O(logn)). The main unattractive feature of this case is that the resulting composite graph will have very low (asymptotic) reliability, although the rate of decrease may be quite slow in n.

Finally, in the preceding sections we have restricted our algorithms to compute connectivity probabilities. In fact, similar results will hold in case the building block contains a certain distinguished subset of nodes K, and we are interested in the probability that all the copies of K remain connected. The corresponding lattice-theoretic tools are described in [4].

* * * * *

References

[1] A. Agrawal and A. Satyanarayana, An O(|E|) Time Algorithm for Computing the Reliability of a Class of Directed Networks, Operations Research 32 (1984) 493-526.

[2] M.O. Ball, The Complexity of Network Reliability Computations, Networks 10 (1980) 153-165.

[3] D. Bienstock, Large-Scale Network Reliability, Ph.D. Thesis, MIT Operations Research Center (1985).

[4] D. Bienstock, Some Lattice-Theoretic Tools for Network Reliability Analysis, Working paper #21-85-86, GSIA, Carnegie-Mellon University (1985).

[5] D. Bienstock, An Algorithm for Reliability Analysis of Planar Graphs, Working paper #20-85-86, GSIA, Carnegie-Mellon University (1985).

[6] J.A. Buzacott, A Recursive Algorithm for Finding Reliability Measures Related to the Connection of Nodes in a Graph, Networks 10 (1980) 311-327.

[7] L. Comtet, Advanced Combinatorics, D. Reidel Publishing Co., Doldrecht, Holland (1974).

[8] A.K. Kel'mans, Connectivity of a Probabilistic Network, Automation and Remote Control 3 (1967) 444-460.

[9] L. Fratta and U. Montanari, Decomposition Techniques for Evaluating Network Reliability, IBM report RC5923 (1976).

[10] A. Rosenthal, Computing the Reliability of Complex Networks, SIAM J. Appl. Math. 32 (1977) 384-393.